**Библиографические ссылки**

1. СЭД (Рынок России). М. : Аналит. агентство TAdviser, 2012 [Электронный ресурс]. Режим доступа: http://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%A1%D0%AD%D0%94_(%D1%80%D1%8B%D0%BD%D0%BE%D0%BA_%D0%A0%D0%BE%D1%81%D1%81%D0%B8%D0%B8), свободный.

2. Об утверждении Административного регламента предоставления Министерством связи и массовых коммуникаций Российской Федерации государственной услуги по подтверждению подлинности электронных цифровых подписей уполномоченных лиц удостоверяющих центров в выданных ими сертификатах ключей подписей : приказ № 360 от 28.12.2011 г.

3. *Домарев В. В.* Безопасность информационных технологий. Системный подход. Киев : ООО ТИД Диа Софт, 2008. 992 с.

4. ГОСТ Р ИСО 7498-2-99. Информационная технология. Взаимосвязь открытых систем. Базовая эталонная модель. Ч. 2 : Архитектура защиты информации. М., 1999.

5. Контроль версий с использованием Git. М. : Региональный финан.-экон. ин-т: Бизнес-школа информационных технологий [Электронный ресурс]. Режим доступа: http://it.rfei.ru/~13, свободный.

# ТИПОВЫЕ АТАКИ НА ВЕБ-СИСТЕМЫ И СПОСОБЫ ЗАЩИТЫ ОТ НИХ В ВЕБ-ФРЕЙМВОРКЕ DJANGO

*Ю. А. Макаров*
(Тюмень, ТюмГУ, winnerer@ya.ru)

В работе рассмотрены наиболее популярные уязвимости и атаки, а также способы защиты от них в веб-фреймворке Django. Данная работа будет интересна разработчикам, менеджерам, руководителям, инвесторам веб-проектов при выборе веб-фреймворка при рассмотрении аспектов безопасности и надежности разрабатываемого продукта.

**Цель работы**

Целью работы является обзор наиболее типовых уязвимостей веб-серверов а также способов защиты, применяемых на практике, в том числе в веб-фреймоврке Django.

**Введение**

Несмотря на то, что возможность проведения таких типов атак, как Cross Site Scripting (XSS) и Cross-Site Request Forgery (CSRF) на различные веб-сервера была сформулирована еще в 1988 г., доля подверженности сайтов к ним остается большой и даже демонстрирует рост в последние годы. Согласно исследованию Positive Technologies, детально протестировавших 123 сайта, было обнаружено 1817 уязвимостей различной степени риска. В данной работе будут рассмотрены наиболее популярные из них а также средства и методы, применяемые в фреймворке Django для их недопущения.

Большинство веб-разработчиков предпочитают язык программирования PHP для разработки веб-приложений ввиду его распространенности и простоты использования. Тем не менее, если поставленные условия задачи не подразумевают явное использование данного языка, то применение языка Python и веб-фреймворка Django является достойной альтернативой применению PHP.

Использование Python в качестве языка для веб-программирования имеет ряд преимуществ:

- Отсутствие проблемы с кодировками – все строки являются Unicode строками (Python 3, в Python 2 есть ascii строки и Unicode строки);
- Модули;
- Множественное наследование;
- Объектно-ориентированный подход;
- Многопоточность (ввиду использования GIL – Global Interpreter Lock – данные потоки не являются полноценными потоками);
- Понятность кода.

**Обзор атак на веб-сервера**

Самой популярной атакой, по данным Positive Technologies на 2011 г., является атака Cross-Site Request Forgery. Заключается

она в следующем. Допустим на вашем сайте есть форма, в которую вводится некоторая информация и отправляется на ваш же сайт посредством GET-запроса. Атакующий же создает некоторую страницу в Интернете (или присылает вам на почту, вариантов достаточно много), на которой прячет под видом обычной информации (или страницы об ошибке) некоторый скрипт или картинку со специально сформированной ссылкой (атрибут src), который посылает запрос на вашу страницу, выполняя тем самым действия от вашего имени на вашем сервере.

Решением данной проблемы может показаться смена метода GET на POST, но в этом случае злоумышленник не сможет лишь использовать атрибут SRC-картинки для незаметного запроса, а javascript (если вы его не выключили) останется в силе.

Другим предположением о защите от подобного рода уязвимости может стать проверка HTTP_REFERRER – заголовка, который указывает на источник, откуда пришел запрос. Но, во-первых для конфиденциальности его отправку можно отключить в браузере, а во-вторых, существуют скрипты, которые позволяют скрыть Referrer.

Защита от данной уязвимости достаточно простая и заключается в хранении дополнительной переменной в cookie, а также в создании дополнительного поля в вашей форме. В терминах Django его обычно называют csrfprotect. При отправке на сервер переменная cookie и значение этого поля сравниваются, в случае их идентичности выполняются необходимые действия, а в случае несовпадения выдается соответствующая ошибка.

По идеологии Django любая POST-форма изменяет соответствующий ее парамерам объект, поэтому дополнительно к полям формы добавляется поле csrfprotect. При этом код шаблона выглядит примерно следующим образом

```
<form method="POST"
  {% csrf_token %}
  {{form}}
  <input type="submit"
</form>
```

Следующей по популярности является уязвимость утечки информации (Information Leakage), найденной на 52 % проверенных сайтах. Этот тип уязвимости возникает в тех случаях, когда сервер публикует важную информацию (например, комментарии разработчиков или сообщения об ошибках), которая может быть использована для компрометации системы.

Во фреймворке Django существует строгое разграничение между средой для разработки и тестирования веб-сервиса и его работой на «боевом» сервере. За это отвечает специальная опция в главном файле конфигурации проекта DEBUG, которая служит для активации режима отладки. При этом гарантируется, что пользователь не получит отладочной информации в случае, если она деактивирована. Если же все-таки неприятный инцидент произойдет, то в зависимости от веб-сервера, из-под которого запущен данный фреймворк, пользователь получит соответствующую ошибку (чаще всего Internal Server Error 500, если не установлено иное).

Примерно на том же уровне распространенности находится уязвимость Brute Force – 52 %. Данный тип уязвимости известен всем и заключается в банальном полном переборе каких-либо данных (чаще всего учетных, таких как логин и пароль) по словарю или всем возможным комбинациям.

К сожалению, в Django нет механизмов, препятствующих данному типу атаки. Разработчики советуют использовать соответствующие плагины для веб-серверов, из-под которых ведется запуск фреймворка. Также есть сторонние приложения, которые подключаются к Django и реализуют соответствующую функциональность такими методами, как отслеживание количества ввода информации в соответствующие поля и использующие различные методы, вроде CAPTCHA (от англ. Completely Automated Public Turing test to tell Computers and Humans Apart – полностью автоматизированный публичный тест Тьюринга для различения компьютеров и людей). Подключение сторонних модулей возможно благодаря модульности фреймворка – даже ваши собственные наработки являются подключаемыми модулями к основному проекту, поэтому это не является сложным.

На следующем месте по популярности стоит уязвимость внедрения SQL (SQL Injection). Данная уязвимость, согласно тому же исследованию Positive Technologies, наиболее популярна на PHP. Зачастую разработчики данного языка программирования забывают экранировать и проверять типы данных, поступающих от пользователя, благодаря чему данная уязвимость очень популярна. Однако во фреймворке Django версиях 1.1.× (до 1.1.4) и 1.2.× (до 1.2.5) была также уязвимость, связанная с недостаточной фильтрацией CSRF-cookie, благодаря которой злоумышленник мог получить доступ к чувствительной информации или выполнить произвольный программный код.

Что же касается SQL Injection и Cross-Site Scripting (XSS), занимающие 47 % и 40 % соответственно, то в Django используется ORM, которая преобразует запросы к базе данных а также запрашиваемую информацию в объект таким образом, что для получения необходимых значений необходимо всего лишь обращаться к ее соответствующим свойствам. При этом экранирование данных производится автоматически, как и составление SQL-запросов. Проверку типов входных данных берут на себя формы – специальные объекты, формирующие HTML-форму, которая отдается пользователю, и считывающие введенные в нее данные.

XSS чаще всего встречается в тех местах, где уделено недостаточно внимания экранированию различных символов, используемых для выполнения кода на стороне клиента (чаще всего браузера). Например в качестве параметра в форму ввода текста передается скрипт, который по алгоритму выводится на страницу пользователя или администратора. В случае, если этот текст не был экранирован, то скрипт может производить любые манипуляции с DOM-деревом у клиента, а также с его cookies. Встречаются также и более сложные XSS-атаки, такие как внедрение кода в векторную графику (SVG) и неверную интерпретацию подобного изображения браузером. Последствия внедрения подобного скрипта на посещаемый ресурс, думаю, достаточно понятны.

**Заключение**

Если следовать концепции разработки веб-фреймворка Django и его документации, писать тщательно обдуманный и хорошо спроектированный код, то большая часть популярных уязвимостей в проектах, основанных на нем, не работает даже в базовой настройке. Однако для тех уязвимостей, которые не решает Django, существует множество сторонних компонентов, которые позволяют уберечься от атак подобного рода.

## ПРИМЕНЕНИЕ MIDI-ФОРМАТОВ В КАЧЕСТВЕ СТЕГАНОГРАФИЧЕСКИХ КОНТЕЙНЕРОВ

*С. А. Неймышева*
(Екатеринбург, УрГУПС)

Современный прогресс в области глобальных компьютерных сетей и средств мультимедиа привел к разработке новых методов, предназначенных для обеспечения безопасности передачи данных по каналам телекоммуникаций и использования их в необъявленных целях, и имеет большое значение для развития автоматизированного рабочего места (АРМ) сотрудника. В связи со всеобщей компьютеризацией появилась необходимость использования планшетных ПК в рабочих целях. Так как планшетные ПК позволяют работнику быть мобильным, то очень часто он находится вдали от рабочего места, тем самым ограничивается доступ к определенным рабочим ресурсам. К примеру, для получения одноразового пароля используют программы, не защищенные от перехвата пароля. Эту проблему могут решить методы стеганографии, которые скрывают сам факт передачи сообщения, используя стегоконтейнеры. Эти методы, учитывая естественные неточности устройств оцифровки и избыточность аналогового видео- или аудиосигнала, позволяют скрывать сообщения в компьютерных файлах (контейнерах).

В настоящее время методы компьютерной стеганографии развиваются по двум основным направлениям: